Summary of the CMS DST Workshop Napoli, June 16-18, 2004

Norbert Neumeister

CPT Joint Technical Board, June 23, 2004

Introduction

- Goal of the workshop:
  - Feed-back on current DST concept and implementation
  - Review of CMS Framework for Analysis and Reconstruction:
    - Use cases, use experience, comparison with others
    - Data Structures
    - Access Methods
    - Relationships with other domains
  - Develop: short-term (weeks) and long-term plan
- Excellent organization thanks to Luca Lista;
- ~30 people from PRS and CCS attended the workshop
- Not too much feed-back from PRS DST users yet
- A lot of discussion: positive spirit!
- Outcome:
  - Short-term development (but need to define timescale);
  - Some progress, still more to clarify; discussion and WORK is not over
- My summary may be biased!

Agenda

Experience with COBRA, ORCA, FAMOS	
Experience from E/gamma	D. Futyan
Experience with Ecal-calibration stream	P. Meridiani
Experience from Tracker b/tau	W. Adam
CMS Event Data Model	
Introduction to CMS Event Data Model	N. Neumeister
CARF Scenarios	V. Innocente
Overview of EDMs	M. Paterno
Keeping track of the data	
Introduction	L. Silvestris
RecConfig Mechanism	T. Todorov
MetaData Model	V. Innocente
MetaData Model from D0/CDF discussion	M. Paterno
Experience and Development	
Introduction	T. Todorov
Persistency of polymorphic components	L. Lista
Planning for the future	
Summary	V. Innocente

PRS Experience (I)

- CaloRecHits
  - should become RecObj
- EcalPlusHcalTower
  - provide links back to RecHits (TRecRef)
  - apply thresholds?
- No need to store calorimeter clusters
- Electron candidates
  - Refit electron tracks, brem recovery, etc. possible on DST?
- Ecal calibration:
  - Needs further selection: DST → mini DST (today mini DST is Root Tree)
  - Can the EgammaCalib object be reused for other purposes?
  - Work on software for mis-calibration is underway
- ORCA startup time must be improved!

PRS Experience (II)

- Store additional info with RecMuon to allow more flexible isolation algorithms
- RecAlgorithms are too heavy for simple selection algorithms
- Regional reconstruction:
  - HLT
  - SubEvent vs. Region
- RecAlgo documentation → automatic (Configuration DB)
- Reading DSTs:
  - (default) Configuration must be stored with a dataset!
  - Revisit COBRA plugin-in manager
  - Today: dependency on order of algorithms defined in BuildFile
- Missing: How to store event-by-event properties (Collections?)
- Value semantics for RecCollections?
  - TTrack vs. RecTrack
- Data access
  - Meta data model
  - Meta data stored with event data
- Tools to copy datasets to a remote site

Event Data Model

- An Event Data Model (EDM) provides a mechanism for managing data related to an event (crossing) within a program.
- An EDM is not:
  - a persistency mechanism;
  - an I/O mechanism;
  - a file format
- An EDM allows for independent development of data objects and algorithms
- **Key issues** are (experience from other experiments):
  - **Event class:** collection of data for one event (crossing)

    Is the CMS event class ok?
  - **Navigation**: efficient location of specific *pieces* and association between *pieces* of the event
  - **Schema evolution:** backward compatibility
- How many levels: Raw data → ESD/DST → AOD ?

Event Data

- We store:
  - MCinfo: MC generator (HEPEVT)
  - GEANT: SimTrack, SimVertex, SimHits (sub-detector specific)
  - Digis (raw data)
  - Associations
  - Level-1 trigger info
- Persistent analysis objects
  - **Event reconstruction** is very CPU intensive
  - Store the result of **event reconstruction**
  - Provide **compact** information for analysis

➢ **Implications:** data access, data distribution, analysis model

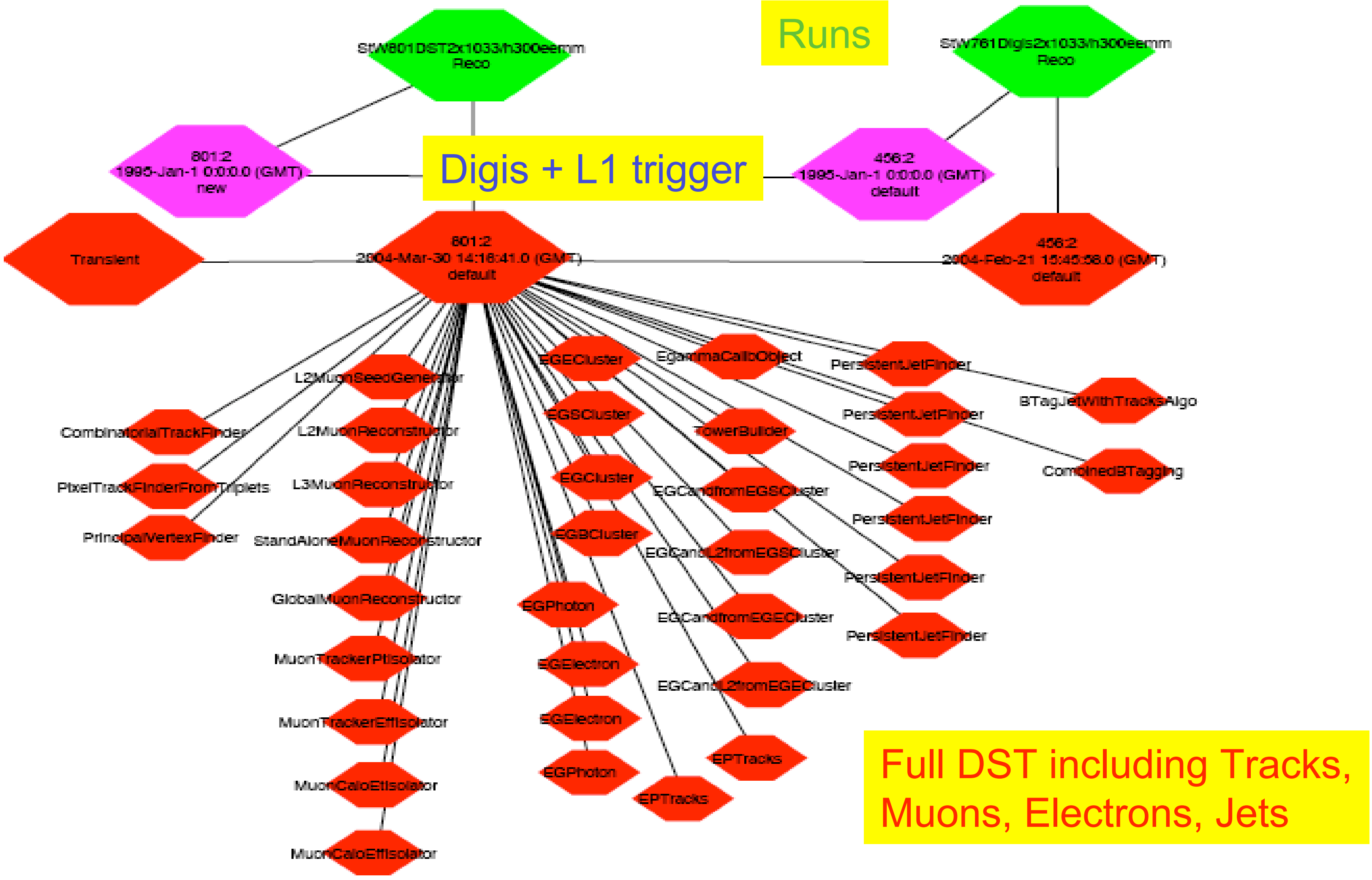
What is an ESD/DST

- Store a complete record of all objects created during reconstruction

- Organized in collections: RecCollection
  - Uniform user interface for all reconstructed objects
  - Today: In total about 50 different RecCollections
  - Today: 150 - 250 KByte/event

- Reconstruction is "expensive"
  - Reconstruct events only few times (not at user level)
  - Reconstruction is done "on demand"
  - Avoid access to RAW data as much as possible
  - Allows to re-reconstruct objects

DST



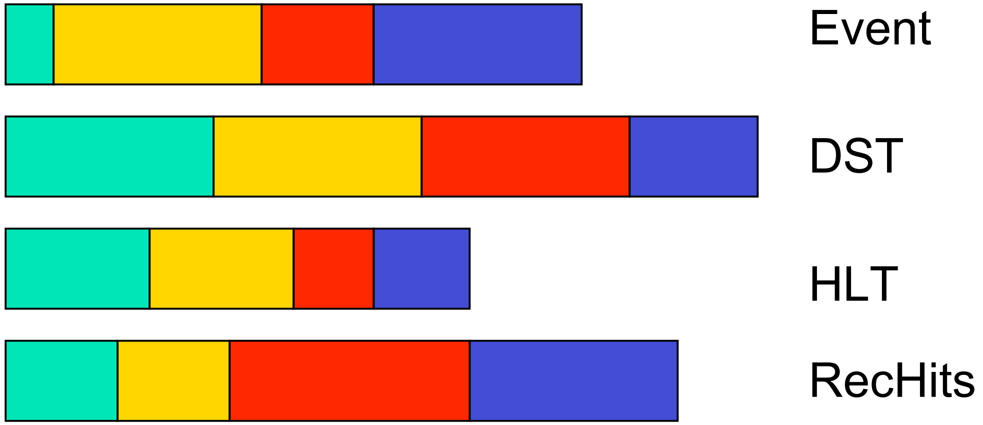
Runs

Digis + L1 trigger

Full DST including Tracks, Muons, Electrons, Jets

Data Organization

Mapping objects to files: *n* objects, *m* files



- File size
- How many files
- Clustering

COBRA provides:

- a flexible way to map objects to files
- user interface to select objects to be stored

How many files must be copied to read a single event?
Needs to be optimized!

Data Size

- 500 events: H → eeμμ; $m_H = 300$ GeV
- Low luminosity (pile-up)

14.2MB	EVD1_MCInfo.1.h300eemm.SimHits
0.5MB	EVD1_Collections.1.h300eemm.Digis
0.9MB	EVD0_Events.1.h300eemm.Digis
200.6MB	EVD2_Digis.1.h300eemm.Digis
121.3MB	EVD3_Assocs.1.h300eemm.Digis
0.4MB	EVD4_L1Trigger.1.h300eemm.Digis
0.8MB	EVD1_Collections.1.h300eemm.DST
8.3MB	EVD0_Events.1.h300eemm.DST
76.7MB	EVD1_DST.1.h300eemm.DST
0.6MB	EVD2_DSTMC.1.h300eemm.DST
0.3MB	EVD4_HLT.1.h300eemm.DST
34.5MB	EVD5_RecHits.1.h300eemm.DST
8.3MB	EVD0_Events.1.h300eemm.DST

DST Issues

- We have a technology to store **collections** of **reconstructed objects** in a consistent way
- A mechanism for event-by-event properties is missing
- Association between high-level objects and MC is missing
- Mapping of RecObjects into files not optimized
  - how many files writeDST should produce?
- Mechanism to combine objects of different types is currently missing
- We should store HEPEVT info in separate file(s)
  - not together with GEANT information
- Store compact/selected GEANT (SimTrack/SimVertex) information in DST

- Use current DST model to:
  - Produce MasterDST: store all reconstructed objects of an event
  - Data skimming: mini/micro DST, derived analysis specific DST
  - Provide tools to create mini/micro DSTs (analysis/group specific); optimize size and contents

Reading DSTs

- Need COBRA/ORCA software environment
  - To read and interpret data
  - How may file one needs to open to read one event?
    - needs optimization!
  - Heavy system (compared to ROOT)
  - Slow start-up (compared to ROOT)
  - No read-only mode yet (COBRA/ORCA Light)
  - Run ORCA to:
    - Fill histograms
    - Copy DST objects into Root Trees
  - Heavy dependency on BuildFile (need to specify all algorithms used to produce the DST in BuildFile) → wrong concept
  - **DST read-only mode:** only libraries of RecObjects need to be specified (loaded)
  - If ORCA is also Analysis tool → faster startup!

Beyond ESD/DST

- Do we need a further layer: AOD?
  - Define requirements (possibility to refit, link to original objects, etc.)
  - Provide a way to read data independent of CMS software environment (ntuples, root trees, etc.)?
  - How to guarantee the correct interpretation of data?
- Doesn't require access to detector code in using it:
  - BUT still needs COBRA
- **Collect requirements before working on prototype!**
- CMS data are already stored using ROOT I/O
  - Why should be store it twice?
  - What are the advantages of Trees compared to our current event structure?

Data Tiers and Event Model

- A-priori
  - Object clustering
  - Event clustering (streams)
- A posteriori
  - Skimming
  - Selective cloning (pruning)
  - Re-clustering

Analysis

- The question is not if we are going to use ROOT but "How we are going to use ROOT"
  - A good fraction of CMS physicists will use ROOT as analysis tool
- Do we need a data format which allows analysis without COBRA/ORCA software? (Ntuples)
- COBRA support for ROOT needed:
  - COBRA plug-in to ROOT (ROOT binding)
  - Who provides "event loop"
- Dictionary: SEAL vs. ROOT
  - No duplication!
  - We do not need a dictionary for all classes!
- Provide/request POOL - ROOT interface:
  - ROOT should be able to read POOL data
- Avoid code duplication at analysis level

Meta Data

- Traditionally Metadata are data that describe other data
  - schema, protocols, type-dictionaries
- Complexity requires coherent access mechanisms
- Navigation is essential for an effective physics analysis
- Event-Collection Meta-Data
- Environmental data
  - Detector and accelerator status
  - Calibration, alignment (luminosity, selection criteria, …)
- Event Data, User Data

- Which framework should handle Metadata?

Configuration (MetaData)

- Different level of Configuration
  - Generator configuration (today only in RefDB)
  - Geometry configuration (today in RefDB and in cvs using tag; release procedure in order to identify the geometry used)
  - Magnetic Field same as Geometry
  - Simulation: configuration is stored inside META Data files
  - Reconstruction are stored inside META Data files
    - DST configuration (RecConfig)
    - Mini-DST (RecConfig)
  - Calibration and mis-alignment
- Granularity for Configuration (MetaData)
  - Event collection (Run) level
  - Event level
  - Sub-event level

Distributed Analysis

- User-friendly interface for physicists (Data access)
  - **Input:** Query for a consistent collection of events (DataSet + owner concept) with a set of configuration parameters
  - **Output:** Data location (T1-T2) and event collection xml catalog that could be used from COBRA/ORCA
- Data Clustering and re-clustering
  - This is possible in CMS model
  - Challenging task and essential for DA
  - **Requirement:** Starting from an event collection (stream) build a new event collection that contains only the relevant part for a specific analysis and publish data.

Meta Data Future

- Configuration should be moved in its own database
  - Is it the same as the Condition DB?
  - How we identify versions and variants?
  - Should we refer to configuration items by a unique id or through its attributes?
- Owner, Originator, Transformation, Dataset
  - Do we need to distinguish between these concepts?
  - What is the relationship among them and w.r.t. the configuration?
- Naming policy
  - Can we afford multiple naming policies?
  - At which level naming policies should be enforced?
  - Can we really implement a unique consistent naming policy in a fully distributed environment?

Development

- Short-term
  - Find solutions to existing problems with the current DST
  - Consolidation of Reco and Meta Data
  - In order to make the next DST iteration more useful for PRS groups
  - Time scale?
  - Modifications in CARF needed (who?, when?)
  - Modified/new ORCA algorithms should be available a couple of weeks after
- Long-term
  - Add new functionality
  - AOD
  - ROOT binding
  - New configuration
  - Calibration

Short Term (I)

- Consolidation of Reco and MetaData
  - Regional reconstruction and nested algorithms
  - Persistency of default RecConfig!
  - Support for read-only mode (ORCA Light)
    - without access to the reconstruction code
  - Access to Meta Data required to read DST
- Configurable "free" (i.e. not RecAlgorithm) algorithm
  - For usage as component of RecAlgorithm
  - RecAlgorithm wrapper around it
- Regional reconstruction
  - Region vs. Context
- Access to and persistency of results of regional reconstruction
- Filtered collections with filter parameters in RecConfig
- Conflicts/Ambiguities between RecObjects from different collections
- Monte Carlo DST and association from/to RecObjects

Short Term (II)

- Proposed solution for
  - regional reconstruction,
  - configurable free algorithms,
  - filtered collections
- Allow "free" algorithm to be a component of a RecAlgorithm
  - The configurability of the RecAlgorithm imposes some constraints
  - **ConfigAlgorithm**
  - Similar mechanism as RecAlgorithm
  - Light-weight registry needed (COBRA)
  - Interface of ConfigAlorithm is not defined by the framework
    - concrete algorithms have interfaces tailored to their functionality
  - ConfigAlgorithm has a reconstruct method that takes a "region"
    - regional reconstruction
  - Ownership of objects created by a ConfigAlgorithm is passed to the caller

Long Term

- Prototype an AOD package
  - Based on current COBRA event model
  - Satisfies the analysis needs of all PRS groups
  - Does not require access to detector code in using it
- Configuration
  - Develop a Configuration Database
    - Algorithm (parameters)
    - Geometry
    - Detector status (conditions)
  - Id, Version, Variants, other attributes: Internal use, Human friendly
  - Leverage experience and existing software: rcp, etc.
- Regional reconstruction
  - Consistent Event view
  - Incremental reconstruction
  - RecCollection pruning: remove unused objects
- Root
  - Used as autonomous analysis tool
  - Prototype "Root binding"